arXiv:1501.06364v1 [cs.GR] 26 Jan 2015

GPU Programming - Speeding Up the 3D Surface Generator VESTA

B. R. Schlei[*]
GSI, Darmstadt, Germany

Abstract

The novel "Volume-Enclosing Surface exTraction Algorith" (VESTA) generates triangular isosurfaces from computed tomography volumetric images and/or three-dimensional (3D) simulation data. Here, we present various benchmarks for GPU-based code implementations of both VESTA and the current state-of-the-art Marching Cubes Algorithm (MCA). One major result of this study is that VESTA runs significantly faster than the MCA.

Introduction

NVIDIAs toolkits (*cf.*, *e.g.*, Ref. [1]) for the development of CUDA®-based software contain, among many other things, example code for an extended version [2] of the original MCA [3]. Here, we compare the performance of this code with our CUDA®- and ANSI-C-based implementation of VESTA [4] on a Linux-based (*i.e.*, openSUSE 13.1) PC with a GeForce GTX 750 Ti graphics card.

In particular, the times that we have measured (*cf.*, Table 1) are averages over 1000 runs each. The measurements start after the data sets have been loaded into texture memory, and they stop after all point coordinates and triplets of point IDs (*i.e.*, triangles) have been computed on the GPU.

Technique	Extended MCA	Marching VESTA	
Mode	**DCED / L**	**DCED / L**	**Mixed / H**
(a) Points	19,218	12,814	15,292
Triangles	6406	6406	11,362
Time (ms)	**1.43(5)**	**1.28(5)**	1.37(4)
(b) Points	6,128,724	4,085,840	4,852,644
Triangles	2,042,908	2,042,908	3,576,516
Time (ms)	**98.5(1)**	**71.3(1)**	75.9(4)
(c) Points	5,566,998	3,699,086	4,346,120
Triangles	1,855,666	1,855,666	3,147,604
Time (ms)	**23.0(1)**	**18.7(1)**	22.4(1)
(d) Points	33,240	22,208	25,894
Triangles	11,080	11,080	18,350
Time (ms)	**0.82(4)**	**0.63(4)**	0.74(4)
(e) Points	13,859,304	9,267,824	11,178,649
Triangles	4,619,768	4,619,768	8,441,610
Time (ms)	**111.2(1)**	**84.4(1)**	94.6(6)

Table 1: Benchmarks for various processed tomographic data sets: for (a) – (c), *cf.*, Ref. [4] and Ref.s therein, (d) Bucky.raw data is a portion of [1], and (e) Happy Buddha VRI file [5]. For the selected isovalues, *cf.*, Fig. 1.

[*] b.schlei@gsi.de

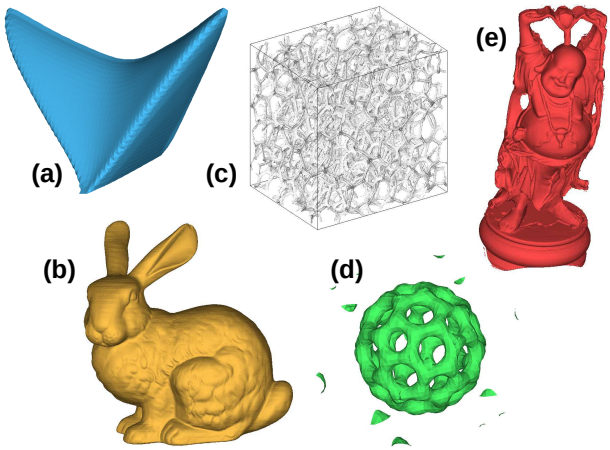

Figure 1: VESTA high resolution "mixed" mode (Mixed/H) isosurface renderings, where the isovalues equal to (a) 139, (b) 150, (c) 135, (d) 128, and (e) 150, respectively.

Results

For the here considered data sets [1, 4, 5], the extended MCA is about (a) 12%, (b) 38%, (c) 23%, (d) 30%, and (e) 32%, *slower* than the marching variant of VESTA [4], when the latter is executed in its low resolution "disconnect" mode (DCED/L). Furthermore, VESTA is also faster even if higher resolution isosurfaces are computed (*cf.*, Fig. 1), which have about twice the number of triangles (*cf.*, Table 1).

Note that the current code implementation of VESTA does *not* yet use parallel streaming, *nor* it does call device kernels from within kernels. As a consequence, further GPU-based code optimisations may result in an even faster VESTA code.

References

[1] NVIDIA® CUDA® Toolkit 6.5; for more detail, *cf.*, https://developer.nvidia.com/cuda-toolkit/

[2] P. Bourke, "Polygonising a scalar field", May 1994; for more detail, *cf.*, http://paulbourke.net/geometry/polygonise/

[3] W. E. Lorenzen and H. E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm", Comput. Graph. 21 (1987), p. 163.

[4] B. R. Schlei, "Volume-Enclosing Surface Extraction", Computers & Graphics 36 (2012) p. 111, doi: 10.1016/j.cag.2011.12.008.

[5] http://graphics.stanford.edu/software/volfill/